Requirements Prioritization Case Study Using AHP

Nancy R. Mead, Software Engineering Institute [vita[1]]

2006-09-23; Updated 2008-09-18 L2 / L[2]

This article describes a tradeoff analysis that can be done to select a suitable requirements prioritization method and the results of trying one method, AHP, in a case study. It is a companion article to the requirements prioritization introduction[3].

The tradeoff analysis and case study were conducted by a team of Carnegie Mellon graduate students under my supervision during a full-time semester-long project [Chung 06][9]. While results may vary from one organization to another, the discussion of how we applied the method should be of general interest.

Identify Candidate Prioritization Methods

For this case study, we considered the Numeral Assignment Technique, Theory-W, and AHP. Each method was ranked according to the factors identified below. The results of the comparison are summarized in Table 1[10].

- Numeral Assignment Technique [Brackett 90[11], Karlsson 95][12]
- Theory-W [Boehm 89[13], Park 99][14]
- AHP [Saaty 80[15]],[Karlsson 96][16], and Karlsson 97a][17]

We briefly discuss AHP, which was selected for this case study.

AHP

AHP was developed by Thomas Saaty and applied to software engineering by Joachim Karlsson and Kevin Ryan in 1997 [Saaty 80][18], [Karlsson 96][19], and [Karlsson 97a][20]. AHP is a method for decision making in situations where multiple objectives are present. This method uses a pair-wise comparison matrix to calculate the relative value and costs of security requirements. By using AHP, the requirements engineer can also confirm the consistency of the result. AHP can prevent subjective judgment errors and increase the likelihood that the results are reliable. There are five steps in the AHP method:

1. Review candidate requirements for completeness.
2. Apply the pair-wise comparison method to assess the relative value of the candidate requirements.
3. Apply the pair-wise comparison method to assess the relative cost of implementing each candidate requirement.

1. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/230-BSI.html (Mead, Nancy)
3. http://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements/545-BSI.html (Requirements Prioritization Introduction)
4. #dsy534-BSI_identify
5. #dsy534-BSI_application
6. #dsy534-BSI_recommendations
7. #dsy534-BSI_ack
8. #dsy534-BSI_refs
9. #dsy534-BSI_Chung
10. #dsy534-BSI_tbl1
18. #dsy534-BSI_Saaty
19. #dsy534-BSI_Karlsson1
20. #dsy534-BSI_Karlsson2

4. Calculate each candidate requirement's relative value and implementation cost, and plot each on a cost-value diagram.
5. Use the cost-value diagram as a map for analyzing the candidate requirements.

Prioritization Method Comparison

We recommend that candidate prioritization methods be compared so that a suitable method can be selected. For this case study, a comparison matrix of desirable features was developed by the student team. We recommend that each organization develop its own matrix of desirable features. The comparison matrix is shown in Table 1[21]. In this article, we have filled in values for the various methods; however, we recognize that this sort of evaluation is subjective, particularly since it was done by students with limited time constraints and no prior experience, so results may vary from one organization to another. Some example evaluation criteria are

- **clear-cut steps:** There is clear definition between stages or steps within the prioritization method.
- **quantitative measurement:** The prioritization method's numerical output clearly displays the clients' priorities for all requirements.
- **high maturity:** The method has had considerable exposure and analysis in the requirements engineering community.
- **low labor-intensity:** A reasonable number of hours are needed to properly execute the prioritization method.
- **shallow learning curve:** The requirements engineers and stakeholders can fully comprehend the method within a reasonable length of time.

Table 1. Comparison of prioritization methods
3= Very Good, 2= Fair, 1= Poor

	Numeral Assignment Technique	**Theory-W**	**AHP**
Clear-cut steps	3	2	3
Quantitative measurement	3	1	3
High maturity	1	3	3
Low labor-intensity	2	1	2
Shallow learning curve	3	1	2
Total score	12	8	16

Application of AHP in the Case Study

We decided to use AHP as a prioritizing method. This was done on the basis of the above comparison, recognizing that the rankings are subjective. Factoring into the rationale behind choosing AHP were the team members' familiarity with the method, its quantitative outputs, and its structure in providing definite steps for implementation. The team followed the five steps of the AHP method to prioritize the security requirements. Three stakeholders were involved in the AHP prioritization process. In this step, the team held one meeting to give instructions and a follow-up meeting to clarify some ambiguous parts of the AHP method.

Review Candidate Requirements for Completeness

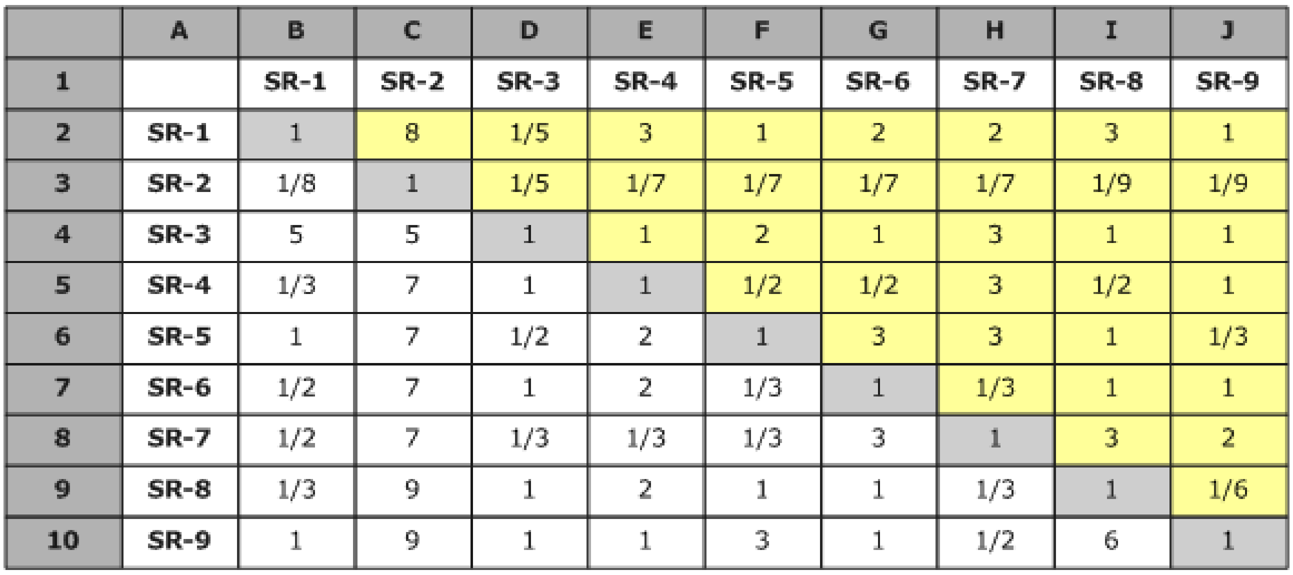
The team reviewed and reanalyzed the security requirements to ensure they were correct, complete, and clear. After meeting with the client, the team revised the requirements based on the feedback received.

21. #dsy534-BSI_tbl1

Apply Pair-Wise Comparison Method

In this step, the stakeholders implemented the pair-wise comparison method of AHP. The team provided brief instructions for using the AHP method to the participants. Because the team generated 9 security requirements, the method should produce a matrix with 81 (9 x 9) cells. However, the participants needed to fill the upper half of the matrix only, and each requirement had a value of "1" when compared to itself. Consequently, each participant had to respond to 36 cells. The team highlighted the cells that required feedback from the participants. A sample of the feedback is shown in the Table 2[22].

Table 2. Prioritization feedback of Acme



	A	B	C	D	E	F	G	H	I	J
1		SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9
2	SR-1	1	8	1/5	3	1	2	2	3	1
3	SR-2	1/8	1	1/5	1/7	1/7	1/7	1/7	1/9	1/9
4	SR-3	5	5	1	1	2	1	3	1	1
5	SR-4	1/3	7	1	1	1/2	1/2	3	1/2	1
6	SR-5	1	7	1/2	2	1	3	3	1	1/3
7	SR-6	1/2	7	1	2	1/3	1	1/3	1	1
8	SR-7	1/2	7	1/3	1/3	1/3	3	1	3	2
9	SR-8	1/3	9	1	2	1	1	1/3	1	1/6
10	SR-9	1	9	1	1	3	1	1/2	6	1

AHP uses a pair-wise comparison matrix to determine the relative value and cost between security requirements. An arbitrary entry in row i and column j of the matrix, labeled a_{ij}, indicates how much higher (or lower) the value/cost for requirement i is than that for requirement j. The value/cost is measured on an integer scale from 1 to 9, with each number having the interpretation shown in Table 3[23] and Table 4[24].

Table 3. Interpretation of values in matrix

Intensity of Value	Interpretation
1	Requirements i and j are of equal value.
3	Requirement i has a slightly higher value than j.
5	Requirement i has a strongly higher value than j.
7	Requirement i has a very strongly higher value than j.
9	Requirement i has an absolutely higher value than j.
2, 4, 6, 8	These are intermediate scales between two adjacent judgments.
Reciprocals	If Requirement i has a lower value than j

Table 4. Interpretation of costs in matrix

Intensity of Value	Interpretation

22. #dsy534-BSI_tbl2
23. #dsy534-BSI_tbl3
24. #dsy534-BSI_tbl4

1	Requirements i and j are of equal cost.
3	Requirement i has a slightly higher cost than j.
5	Requirement i has a strongly higher cost than j.
7	Requirement i has a very strongly higher cost than j.
9	Requirement i has an absolutely higher cost than j.
2, 4, 6, 8	These are intermediate scales between two adjacent judgments.
Reciprocals	If Requirement i has a lower cost than j

The participants filled in the cells of the prioritization matrix to demonstrate the level of concern expressed for the candidate security requirements.

Determine the Priority of Requirements

In this section, we provide instructions for creating cost-value diagrams based on the Excel spreadsheet shown in Figure 1[25]. The formulas that we used in Excel to calculate our results are mentioned throughout this section.

First, the team filled in the lower half of prioritization matrix based on the participants' feedback from the upper half of the matrix. The team then averaged the data over normalized columns to estimate the *eigenvector* of the matrix, which represents the criterion distribution. To do this, we first computed the sum of the columns in the matrix. We then divided each value in the matrix by the column sum. The output is the normalized matrix shown in Figure 1[26]. (In Figures 1 and 2, columns B through K are presumed to contain the raw user feedback and are omitted for clarity of presentation.)

Figure 1. Normalized comparison matrix

	A	L	M	N	O	P	Q	R	S	T
1		**SR-1**	**SR-2**	**SR-3**	**SR-4**	**SR-5**	**SR-6**	**SR-7**	**SR-8**	**SR-9**
2	**SR-1**	0.1021	0.1333	0.0321	0.2405	0.1074	0.1582	0.1503	0.1806	0.1314
3	**SR-2**	0.0128	0.0167	0.0321	0.0115	0.0153	0.0113	0.0107	0.0067	0.0146
4	**SR-3**	0.5106	0.0833	0.1604	0.0802	0.2148	0.0791	0.2254	0.0602	0.1314
5	**SR-4**	0.0340	0.1167	0.1604	0.0802	0.0537	0.0395	0.2254	0.0301	0.1314
6	**SR-5**	0.1021	0.1167	0.0802	0.1603	0.1074	0.2373	0.2254	0.0602	0.0438
7	**SR-6**	0.0511	0.1167	0.1604	0.1603	0.0358	0.0791	0.0250	0.0602	0.1314
8	**SR-7**	0.0511	0.1167	0.0535	0.0267	0.0358	0.2373	0.0751	0.1806	0.2628
9	**SR-8**	0.0340	0.1500	0.1604	0.1603	0.1074	0.0791	0.0250	0.0602	0.0219
10	**SR-9**	0.1021	0.1500	0.1604	0.0802	0.3223	0.0791	0.0376	0.3612	0.1314

The formula of cell L2 is "= B2/ Sum (B$2: B$10)." To generate the remaining values, drag the cursor from L2 to T10.

To determine the score of each requirement, average the row in the normalized matrix by dividing each row sum by the number of requirements (Figure 2[27]). The formula of V2 is "= Average (L2:T2)." To generate all the values in the row, drag the cursor from V2 to V10.

25. #dsy534-BSI_fig1
26. #dsy534-BSI_fig1
27. #dsy534-BSI_fig2

The score of each requirement is the percentage that the requirement adds to the requirements' total value. In this case, SR-1 composes 2.82% of the requirements' total value.

Figure 2. Scores for requirements

	A	L	M	N	O	P	Q	R	S	T	V
1		**SR-1**	**SR-2**	**SR-3**	**SR-4**	**SR-5**	**SR-6**	**SR-7**	**SR-8**	**SR-9**	**Scores**
2	**SR-1**	0.1021	0.1333	0.0321	0.2405	0.1074	0.1582	0.1503	0.1806	0.1314	0.1373
3	**SR-2**	0.0128	0.0167	0.0321	0.0115	0.0153	0.0113	0.0107	0.0067	0.0146	0.0146
4	**SR-3**	0.5106	0.0833	0.1604	0.0802	0.2148	0.0791	0.2254	0.0602	0.1314	0.1717
5	**SR-4**	0.0340	0.1167	0.1604	0.0802	0.0537	0.0395	0.2254	0.0301	0.1314	0.0968
6	**SR-5**	0.1021	0.1167	0.0802	0.1603	0.1074	0.2373	0.2254	0.0602	0.0438	0.1259
7	**SR-6**	0.0511	0.1167	0.1604	0.1603	0.0358	0.0791	0.0250	0.0602	0.1314	0.0911
8	**SR-7**	0.0511	0.1167	0.0535	0.0267	0.0358	0.2373	0.0751	0.1806	0.2628	0.1155
9	**SR-8**	0.0340	0.1500	0.1604	0.1603	0.1074	0.0791	0.0250	0.0602	0.0219	0.0887
10	**SR-9**	0.1021	0.1500	0.1604	0.0802	0.3223	0.0791	0.0376	0.3612	0.1314	0.1582

Check for Consistency

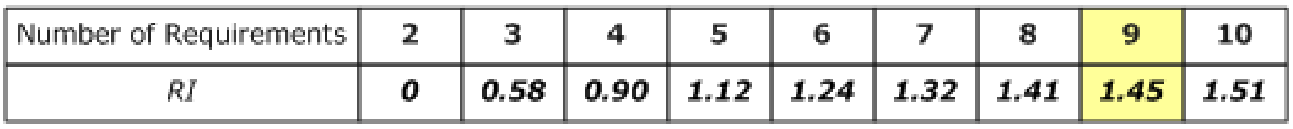
The ability of AHP to test for consistency is one of the method's greatest strengths. The AHP view of consistency is based on the idea of *cardinal transitivity*. For example, if Requirement A is considered to be two times more important than Requirement B, and Requirement B is considered to be three times more important than Requirement C, then perfect cardinal consistency would imply that Requirement A be considered six times more important than Requirement C. In this way, if the participants judge Requirement A to be less important than Requirement C, it implies that a judgmental error exists and the prioritization matrix is inconsistent.

In this section, the team used consistency index/random index (CI/RI) ratio to check the consistency of the results (Figure 3[28]). To compute the CI/RI ratio, the team took the following steps:

1. Calculate the product of the pair-wise comparison matrix and the vector of scores. Make sure that the user data is in decimal form (i.e., "1/5" is now represented as "0.2"). Highlight cells B2 to J10, and type the formula "mmult (B2:J10, V2:V10)." Press Control-Shift-Enter, which applies the formula to the entire highlighted matrix.
2. Calculate the ratios. In cell Y2, calculate the ratio of the score and product values with the formula "=X2/V2" and copy this to the range "Y3:Y10."
3. Calculate the CI value. In cell Y11, calculate the consistency index with the formula "=(average (Y2:Y10) - 9) /8." The value *9* is the number of requirements and *8* is the number of requirements minus one.
4. Calculate the CI/RI score. The RI is the average value of the CI, if the entries in the pair-wise comparison matrix were chosen at random. If the CI/RI score is sufficiently small, then the participants' comparisons are probably consistent enough to be useful. Thomas Saaty suggests that if the CI/RI is smaller than 0.10, then the degree of consistency is satisfactory; however, if the CI/RI is larger than 0.10, inconsistencies exist and the AHP method may not yield meaningful results [Saaty 80[29]]. To calculate the CI/RI score, the team first gets the standard RI value from Saaty's information; a few of those RI values are listed in Table 5[30]. Because the number of security requirements is 9, the RI is 1.45. Second, in cell Y12, calculate the CI/RI score with the formula "= Y11/1.45."

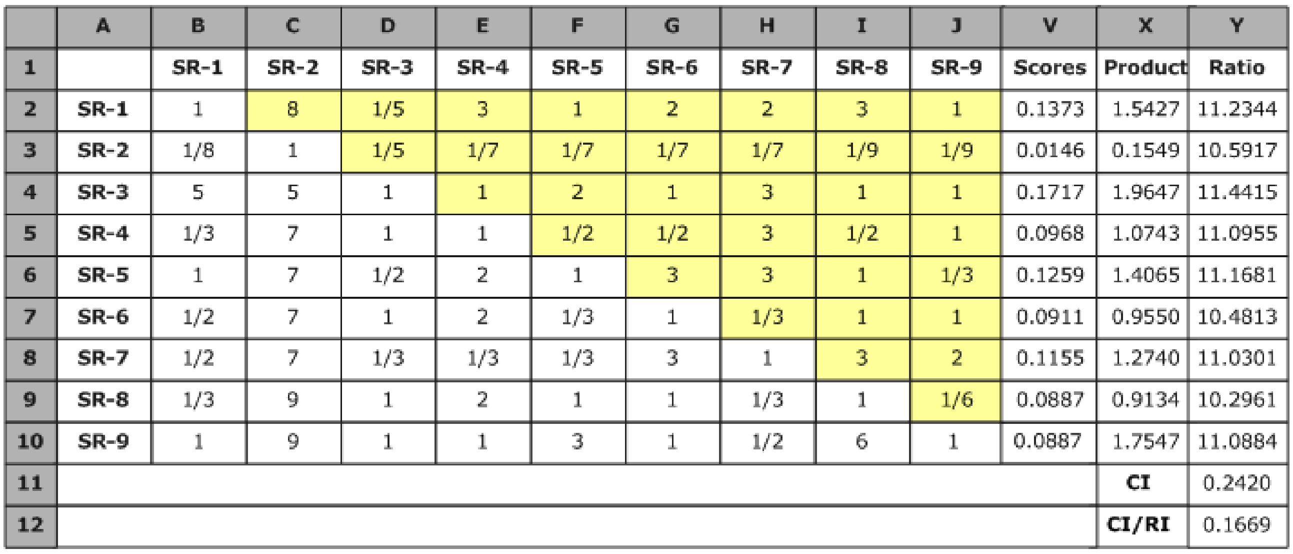
Table 5. Random index values

28. #dsy534-BSI_fig3



Number of Requirements	2	3	4	5	6	7	8	9	10
RI	***0***	***0.58***	***0.90***	***1.12***	***1.24***	***1.32***	***1.41***	***1.45***	***1.51***

Figure 3. Data and results for CI/RI score



	A	B	C	D	E	F	G	H	I	J	V	X	Y
1		SR-1	SR-2	SR-3	SR-4	SR-5	SR-6	SR-7	SR-8	SR-9	Scores	Product	Ratio
2	SR-1	1	8	1/5	3	1	2	2	3	1	0.1373	1.5427	11.2344
3	SR-2	1/8	1	1/5	1/7	1/7	1/7	1/7	1/9	1/9	0.0146	0.1549	10.5917
4	SR-3	5	5	1	1	2	1	3	1	1	0.1717	1.9647	11.4415
5	SR-4	1/3	7	1	1	1/2	1/2	3	1/2	1	0.0968	1.0743	11.0955
6	SR-5	1	7	1/2	2	1	3	3	1	1/3	0.1259	1.4065	11.1681
7	SR-6	1/2	7	1	2	1/3	1	1/3	1	1	0.0911	0.9550	10.4813
8	SR-7	1/2	7	1/3	1/3	1/3	3	1	3	2	0.1155	1.2740	11.0301
9	SR-8	1/3	9	1	2	1	1	1/3	1	1/6	0.0887	0.9134	10.2961
10	SR-9	1	9	1	1	3	1	1/2	6	1	0.0887	1.7547	11.0884
11												CI	0.2420
12												CI/RI	0.1669

Analyze Requirements Using Cost-Diagram Plot

Table 6[31] displays the value and cost CI/RI scores for each participant in the AHP process. As shown in that table, only one CI/RI score is less than 0.10 (the value CI/RI for participant 3). The average CI/RI is 0.16. According to Saaty's determination, then, inconsistencies exist in the results.

To reduce the impact of these inconsistencies, the team decided to delete the largest value in both the value and cost rows and then calculated the average of the remaining two CI/RIs. This refinement resulted in our basing the requirements' value CI/RI on the average scores of participants 2 and 3 and the requirements' cost CI/RI on the average scores of participants 1 and 2.

Table 6. Average costs and value CI/RI scores for participants 1-3

CI/RI Type	Participant 1	Participant 2	Participant 3
Value	**0.25**	**0.16**	**0.07**
Cost	**0.17**	**0.15**	**0.18**

The requirements' final values are shown in Figure 4[32], and the requirements' final costs are shown in Figure 5[33]. The value of each requirement is relative. That is, if the value of a requirement is 20%, this requirement is twice as important as the 10% value of another requirement. The sum of the scores of the requirements should always be 100%. When the value of the requirement is 10%, this requirement consists of 10% of the value of all requirements. The same applies to the cost of each requirement.

According to Figure 4[34], the three most valuable requirements are SR-3, SR-5, and SR-6. Together, they constitute 47% of the requirements' total value. The three least valuable requirements are SR-1, SR-4, and SR-8, which constitute 23% of the requirements' total value. Figure 5[35] shows that requirements SR-4, SR-7, and SR-9 are the three most expensive. Together, they constitute 72% of the requirements' total cost. The

31. #dsy534-BSI_tbl6
32. #dsy534-BSI_fig4
33. #dsy534-BSI_fig5
34. #dsy534-BSI_fig4
35. #dsy534-BSI_fig5

three least expensive requirements are SR-1, SR-2, and SR-3 which constitute 7% of the requirements' total cost.

Figure 4. Value distribution of requirements

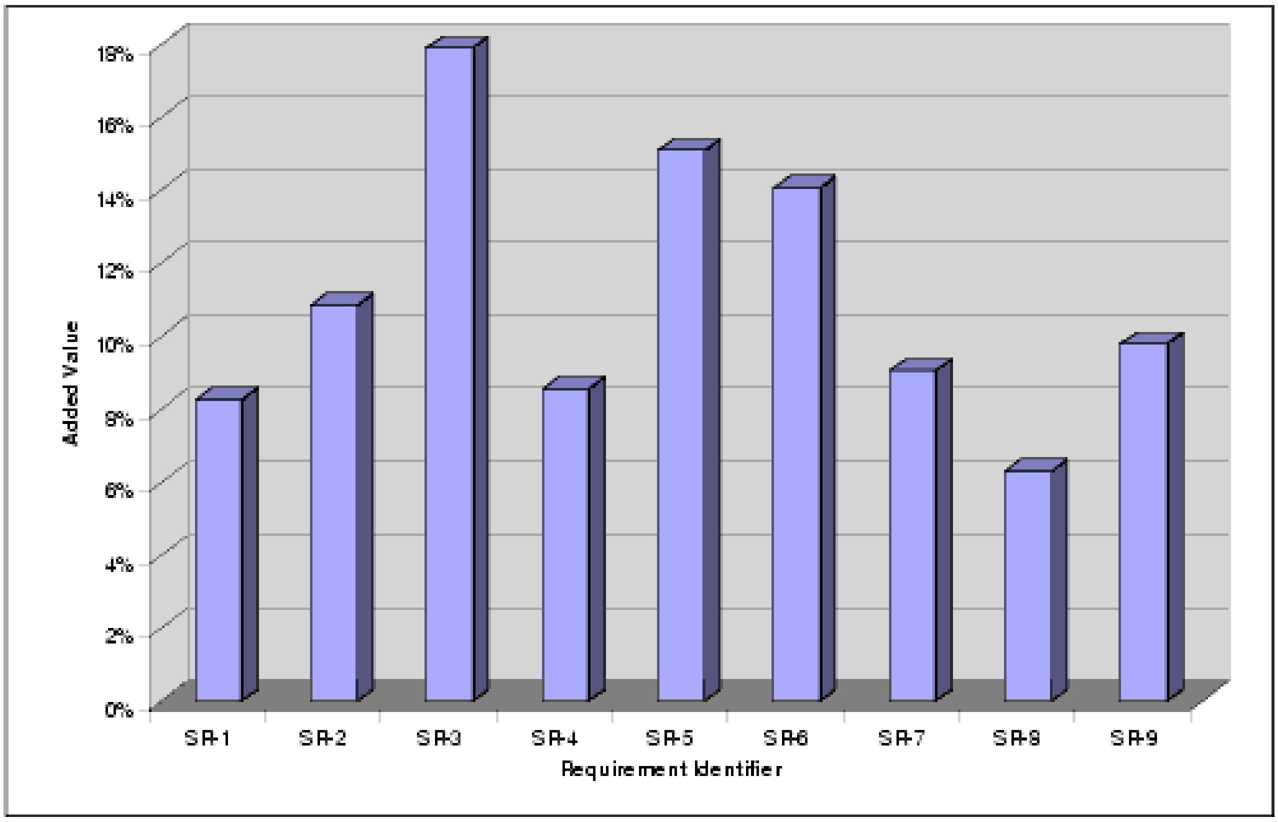
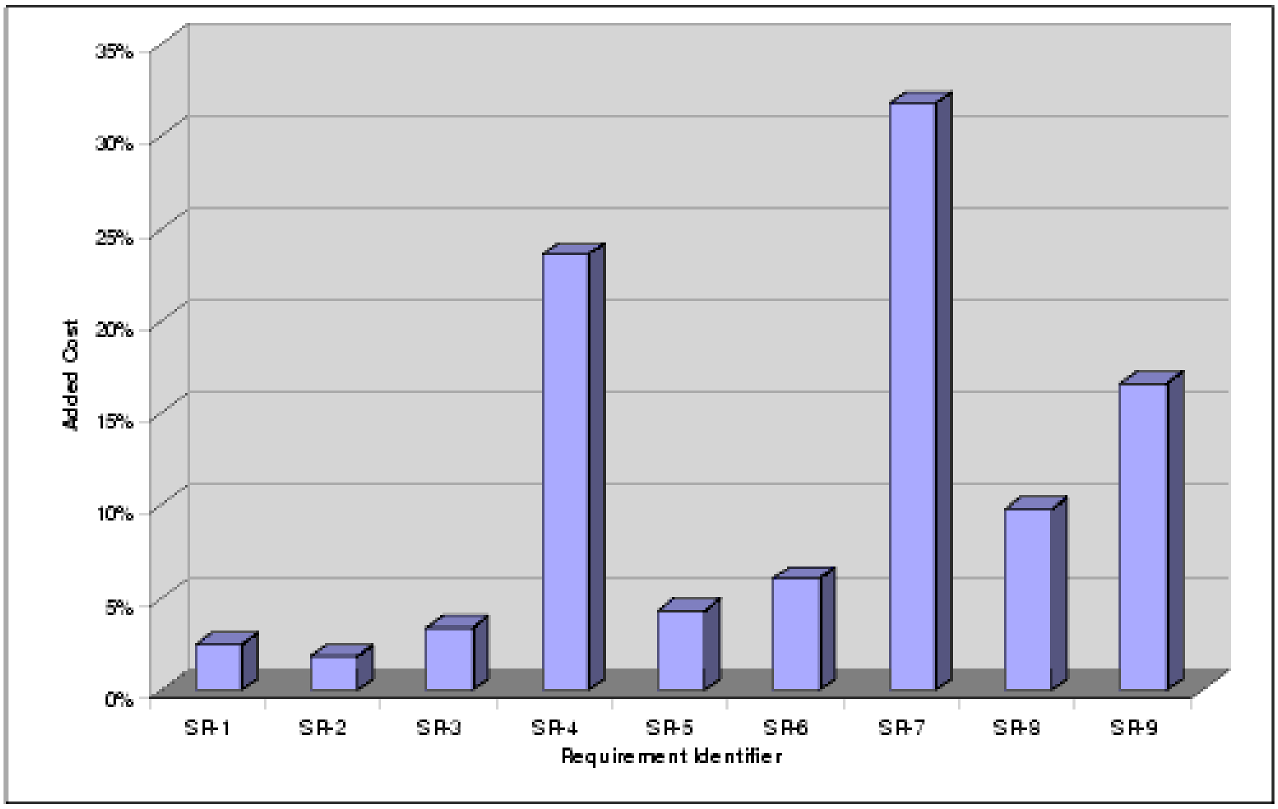


Figure 5. Cost distribution of requirements

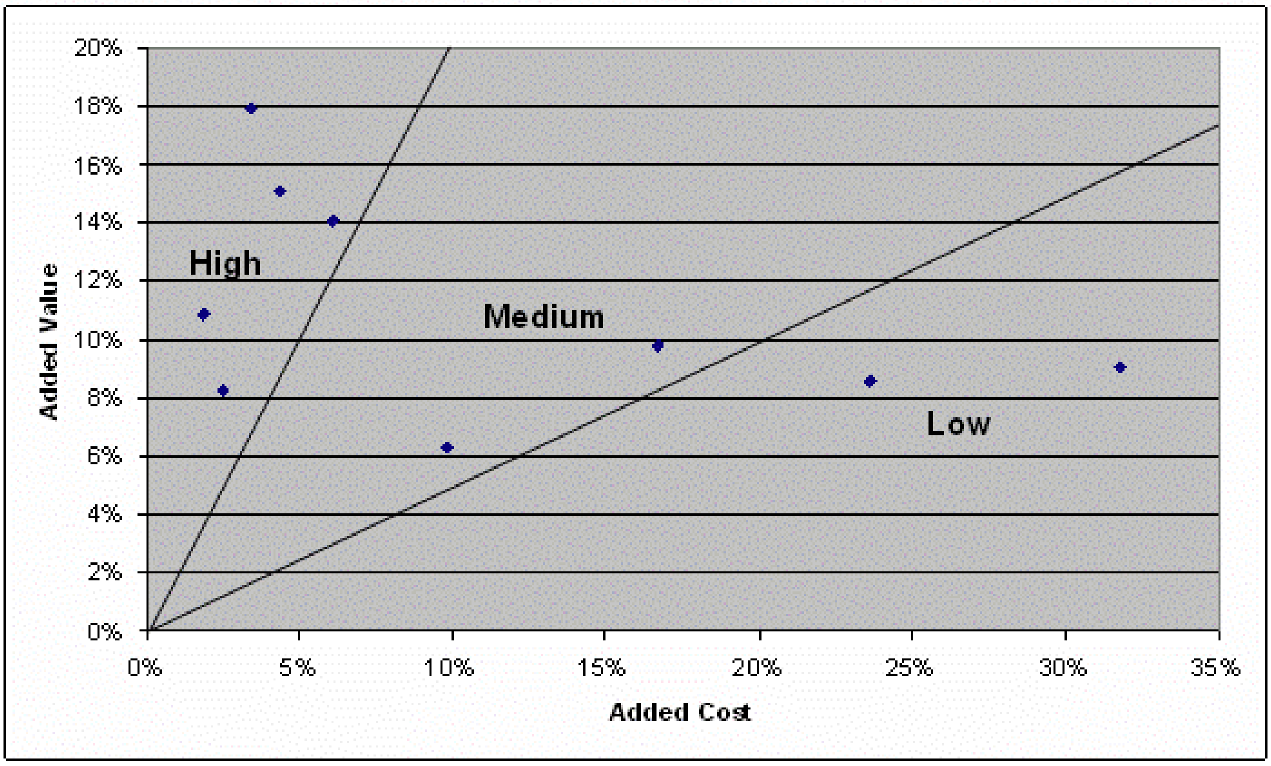


At this point, the stakeholders had assigned a cost and value to each requirement. The next logical step is to calculate the cost-value ratios for each requirement. This way, the stakeholders can pinpoint the requirements that are most valuable and least expensive to implement. The cost-value diagram in Figure 6[36] is divided into three groups:

1. **high** value-to-cost ratio of requirement (larger than 2.0)
2. **medium** value-to-cost ratio of requirement (between 2.0 and 0.5)
3. **low** value-to-cost ratio of requirement (less than 0.5)

Requirements SR-1, SR-2, SR-3, SR-5, and SR-6 are high priority. Requirements SR-4 and SR-7 are low priority. When security requirements are prioritized, the client can implement the security requirements based on their relative priority.

Figure 6. Cost-value diagram of requirements



Reprioritize Security Requirements

During the AHP process, clients were confused with some of the security requirements, and they were not sure about the definitions of *value* and *cost*. For example, some of the clients viewed the costs as the price of implementing these security mechanisms, but other clients thought the costs were the impact of not implementing these security mechanisms.

The team clarified that we were referring to the price of implementing the security mechanisms. The clients then decided to redo the prioritization matrix to get a better result. As a result, the team conducted the AHP process again to generate new prioritization results. The summary results are shown in Figure 7[37] and Figure 8[38].

36. #dsy534-BSI_fig6
37. #dsy534-BSI_fig7
38. #dsy534-BSI_fig8

In Figure 7[39], requirements SR-2 and SR-3 constitute almost half of the value of the security requirements. This means that requirements SR-2 and SR-3 are the most valuable security requirements by far. Compared to the previous prioritization result, the value scores for each security requirement vary.

In Figure 8[40], the results of the cost assessment are very similar to the previous iteration. Apparently requirements SR-4 and SR-7 are the most expensive security requirements to implement.

Figure 7. Refined value distribution of requirements

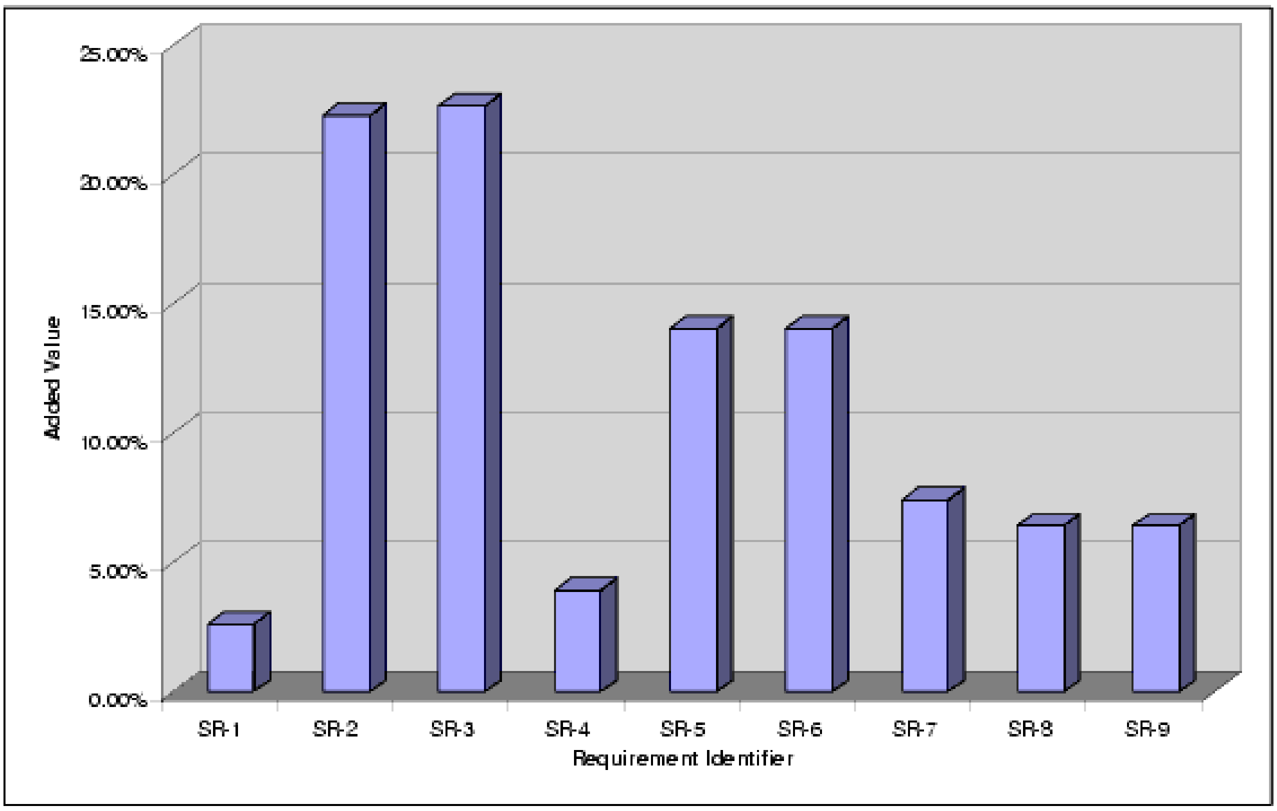
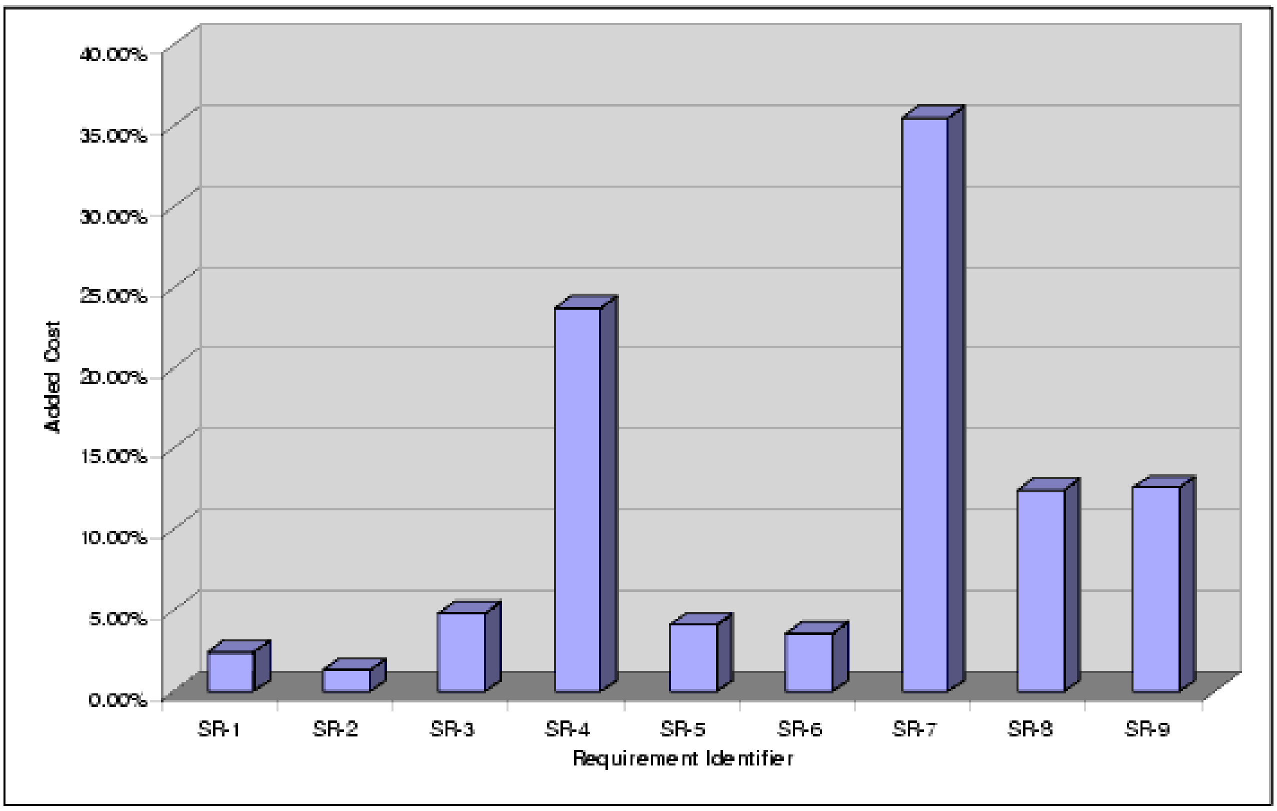


Figure 8. Refined cost distribution of requirements

39. #dsy534-BSI_fig7
40. #dsy534-BSI_fig8

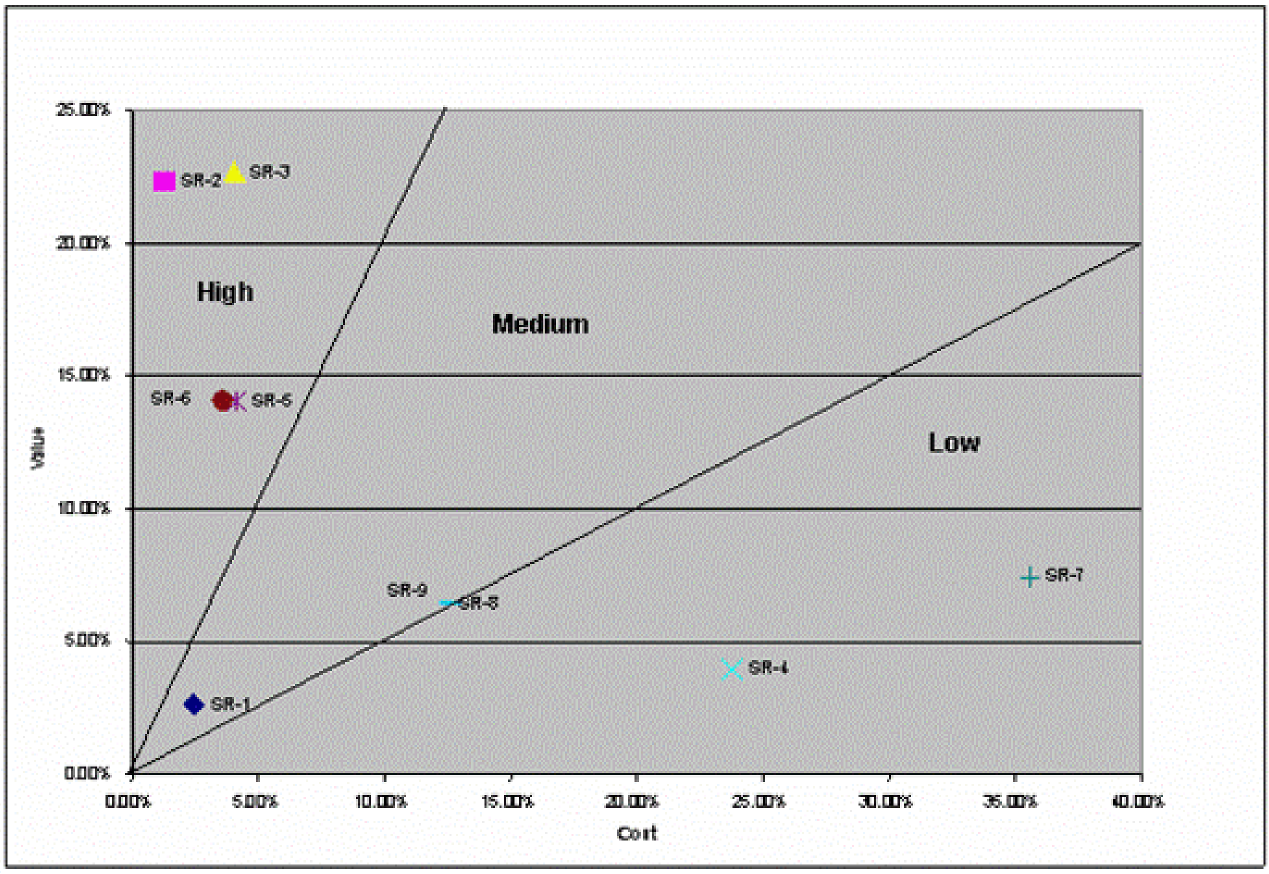


The CI/RI ratios of the value and cost reports are 0.15 and 0.17, which are quite close to the previous results. Although the clients tried to make the new prioritization result consistent, the CI/RI ratios were still larger than 0.10 and judgment errors still exist in the new result.

In Figure 9[41], we see that the client has four security requirements that fall into the high-priority category, three security requirements in the medium-priority category, and two security requirements in the low-priority category. Those requirements with a high value-cost ratio (such as SR-2 and SR-3) fall into the high-priority area. Likewise, those with a low value-cost ratio (such as SR-4 and SR-7) fall into the low-priority area.

Figure 9. Refined cost-value diagram of requirements

41. #dsy534-BSI_fig9



Client Feedback

In informal feedback sessions, we learned that the client would have preferred to see the consistency checker in action because some of the values were derived through negotiation between the developer, administrator, and marketing team. The difference in value perception between the three stakeholder groups was also very interesting.

The client generally found AHP to be clear, easily understood, and able to provide a good indication of the cost/value ratio for prioritizing requirements. However, they would have liked to have known whether cost, value, or both were the main drivers in establishing the priority. Moreover, they thought that evaluating some items was like comparing apples to oranges: two requirements were quite valuable but for very different reasons. Also, the value of the requirement makes the person filling out the survey take many variables into account such as the following:

- frequency of occurrence
- danger of occurrence
- marketability
- system robustness

Different stakeholders may place very different weights on these variables, yet their relative weighting is not taken into account anywhere. It is simply summarized as "value."

The clients felt that the range of values available in assigning to a comparison could be trimmed down to three or four values and their reciprocals.

It is difficult to assess the value of the AHP method with so few requirements to prioritize. It is relatively easy to evaluate a few requirements at the same time. In a small set, too, it is difficult to objectively consider requirements in pairs without considering others as well, turning the matrix completion process into something closer to a ranking.

The client thought that the AHP method needs to be supported by a tool that presents only two requirements at a time to solicit a comparison. As noted below, there is a commercial tool to support AHP, but we were unaware of it at the time. That process should assign only positive integers to the winner. In addition, there should be some cognitive subversion to ensure true responses. One possibility is to trick the user into ranking the pairs multiple times by presenting them in random orders.

Lastly, the clients felt that the consistency-check result is a direct reflection of how the requirements are interpreted. The better defined the requirements are, the more consistent the outcome should be.

Recommendations and Summary

Generally speaking, the AHP method was a straightforward method for prioritizing requirements. However, it was difficult to define the value and cost of each security requirement because the value and cost could be very complicated and could vary dramatically due to the stakeholders' different viewpoints. The Triage approach could be helpful in addressing this problem [Davis 03[42], Davis 05][43]. As it was, each participant had an opinion about the value and cost of the security requirements. For example, a developer may view the value of privacy protection as very low and the cost of privacy protection as very high simply because he or she doesn't feel strongly about privacy issues. On the other hand, a user may think the value of privacy is very high and have no idea about the cost of the technology to ensure it.

In its first attempt to prioritize the requirements, the team asked each participant to prioritize separately so that everyone had a prioritization matrix. Then the team came up with all the participants' scores and averaged those scores. However, prioritization wasn't that easy. Prioritization is an iterative process, and clients repeated the negotiation and consensus process again and again. So, it may not have been a good idea to simply average all the participants' scores. The team recommends that all the participants come together to discuss the priority of the security requirements in a session instead of doing the prioritization individually. During the prioritization process, the stakeholders can ascertain that everyone has the same understanding about the security requirements and further examine any ambiguous requirements. After everyone reaches a consensus, the result of prioritization will be more reliable.

These case studies are part of the Security Quality Requirements Engineering (SQUARE) project [Mead 05][44]. Since these case studies were completed, we have published a report on how to compare SQUARE with other security requirements engineering methods [Mead 07[45]]. We have also published a report examining ways of integrating SQUARE with popular lifecycle models [Mead 08[46]]. We have developed a prototype tool. We have also developed educational materials[47] that can be downloaded.

Recent research has suggested that it may be beneficial to incorporate the results of a threat modeling exercise into the prioritization process, combining the threat modeling results with AHP. This research work is still in progress. We currently plan to extend SQUARE for acquisition and to develop a robust tool that provides both analysis and documentation support.

Note that there are existing tools for some of the methods, such as AHP (see Focal Point[48] and Rational[49], for example) [Karlsson 97b][50].

42. #dsy534-BSI_Davis1
43. #dsy534-BSI_Davis2
44. #dsy534-BSI_Mead
45. #dsy534-BSI_mead07
46. #dsy534-BSI_mead08
47. http://www.cert.org/sse/square/square-description.html
48. http://www.telelogic.com/corp/products/focalpoint/index.cfm
49. http://www-01.ibm.com/software/rational/
50. #dsy534-BSI_Karlsson3

Acknowledgement

This material is extracted and adapted from a more extensive case study report by Carnegie Mellon graduate students Lydia Chung, Frank Hung, Eric Hough, and Don Ojoko-Adams [Chung 06][51].

References

[Boehm 89]	Boehm, B. & Ross, R. "Theory-W Software Project Management: Principles and Examples." *IEEE Transactions on Software Engineering 15*, 4 (July 1989): 902-916.
[Brackett 90]	Brackett, J. W. *Software Requirements*[52] (SEI-CM-19-1.2, ADA235642). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1990.
[Chung 06]	Chung, L.; Hung, F.; Hough, E.;Ojoko-Adams, D. *Security Quality Requirements Engineering (SQUARE): Case Study Phase III*[53] (CMU/SEI-2006-SR-003). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006.
[Davis 03]	Davis, A. "The Art of Requirements Triage." *IEEE Computer, 36*, 3 (March 2003): 42-49.
[Davis 05]	Davis, A. *Just Enough Requirements Management: Where Software Development Meets Marketing*. New York: Dorset House, 2005 (ISBN 0-932633-64-1).
[Karlsson 95]	Karlsson, J. "Towards a Strategy for Software Requirements Selection. Licentiate." Thesis 513, Linkping University, October 1995.
[Karlsson 96]	Karlsson, J. "Software Requirements Prioritizing," 110-116. *Proceedings of the Second International Conference on Requirements Engineering (ICRE'96).* Colorado Springs, CO, April 15-18, 1996. Los Alamitos, CA: IEEE Computer Society, 1996.
[Karlsson 97a]	Karlsson, J. & Ryan, K. "Cost-Value Approach for Prioritizing Requirements." *IEEE Software 14,* 5 (September/October 1997): 67-74.
[Karlsson 97b]	Karlsson, J., Olsson, S., Ryan, K. "Improved Practical Support for Large-scale Requirements Prioritising." *Requirements Engineering Journal 2,* 1 (1997): 51-60.
[Mead 05]	Mead, N. R.; Hough, E.; & Stehney, T. *Security Quality Requirements Engineering (SQUARE) Methodology*[54] (CMU/SEI-2005-TR-009). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2005.

51. #dsy534-BSI_Chung

[Mead 07]	Mead, N. R. *How To Compare the Security Quality Requirements Engineering (SQUARE) Method with Other Methods*[55] (CMU/SEI-2007-TN-021). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, August, 2007.
[Mead 08]	Mead, N. R., Viswanathan, V., Padmanabhan, D., & Raveendran, A. *Incorporating Security Quality Requirements Engineering (SQUARE) into* [56]*Standard Life-Cycle Models*[57] (CMU/SEI-2008-TN-006). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, May, 2008.
[Park 99]	Park, J.; Port, D.; & Boehm B. "Supporting Distributed Collaborative Prioritization for Win-Win Requirements Capture and Negotiation," 578-584. *Proceedings of the International Third World Multi-conference on Systemics, Cybernetics and Informatics (SCI99) Vol. 2*. Orlando, FL, July 31-August 4, 1999. Orlando, FL: International Institute of Informatics and Systemics (IIIS), 1999.
[Saaty 80]	Saaty, T. L. *The Analytic Hierarchy Process*. New York, NY: McGraw-Hill, 1980.

Carnegie Mellon Copyright

1. mailto:permission@sei.cmu.edu